

# A Fully Automated Latent Fingerprint Matcher with Embedded Self-learning Segmentation Module

Jinwei Xu, Jiankun Hu, Xiuping Jia

School of Engineering and Information Technology

The University of New South Wales

Canberra, ACT 2600, Australia

E-mails: jinwei.xu@student.adfa.edu.au; j.hu@adfa.edu.au; x.jia@adfa.edu.au

## Abstract

Latent fingerprint has the practical value to identify the suspects who have unintentionally left a trace of fingerprint in the crime scenes. However, designing a fully automated latent fingerprint matcher is a very challenging task as it needs to address many challenging issues including the separation of overlapping structured patterns over the partial and poor quality latent fingerprint image, and finding a match against a large background database that would have different resolutions. Currently there is no fully automated latent fingerprint matcher available to the public and most literature reports have utilized a specialized latent fingerprint matcher COTS3 which is not accessible to the public. This will make it infeasible to assess and compare the relevant research work which is vital for this research community. In this study, we target to develop a fully automated latent matcher for adaptive detection of the region of interest and robust matching of latent prints. Unlike the manually conducted matching procedure, the proposed latent matcher can run like a sealed black box without any manual intervention. This matcher consists of the following two modules: (i) the dictionary learning-based region of interest (ROI) segmentation scheme; and (ii) the genetic algorithm-

based minutiae set matching unit. Experimental results on NIST SD27 latent fingerprint database demonstrates that the proposed matcher outperforms the currently public state-of-art latent fingerprint matcher.

*Keywords:* Latent fingerprint, fingerprint matching, fingerprint segmentation, dictionary learning, genetic algorithm

## 1 Introduction

The pioneering study for fingerprint identification with its application in distinguishing criminals could be traced back to [1]. The fingerprints inadvertently touched by a person in crime scenes is applicable to identify the criminals or to exclude the suspects. Fingerprints collected from crime scenes are compared to the fingerprints collected from suspects so that the fingerprints belonging to criminals could be identified [2]. As a consequence, the Automated Fingerprint Identification Systems (AFIS) is established and developed to satisfy such urgent need [3]. One important function of the AFIS system is to identify suspects against a large fingerprint database from an unknown fingerprint. There are two basic ways of searching. One approach is fingerprint indexing including classification, where the query fingerprint is mapped into a cluster with similar characteristics and such cluster will become the candidates for further inspection. The second approach is to perform matching on a one-on-one basis against the whole database. In principal, the first approach is most efficient. However, even though some progress has been made on the partial fingerprint indexing [4], few literatures have been found on latent fingerprint indexing. Existing latent fingerprint identification work is virtually on a one-on-one basis which is also our focus in this paper.

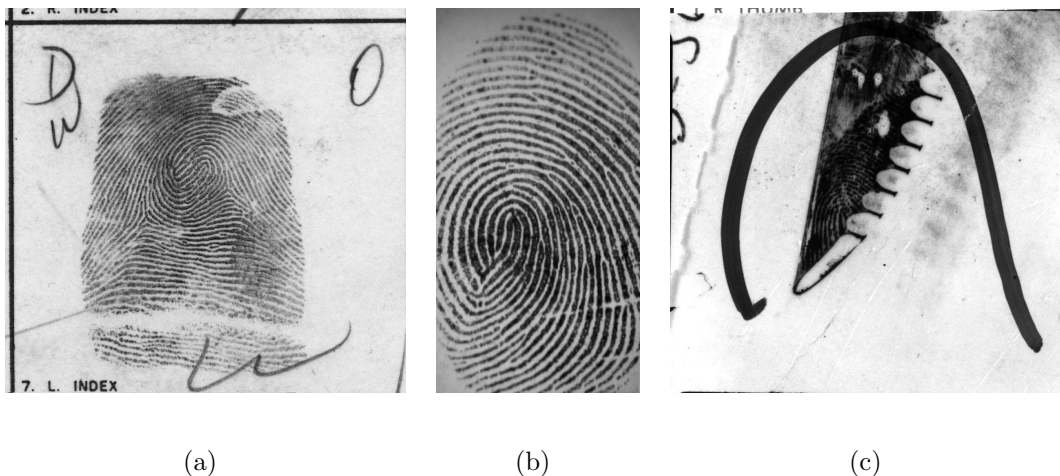


Figure 1: Three types of fingerprints: (a) the rolled print; (b) the plain print; and (c) the latent print.

Therefore in the remaining of the paper, our AFIS discussion is restricted to the category of one-on-one matching unless stated otherwise. AFIS is widely used to identify three main types of fingerprints: the rolled, the plain and the latent. The rolled fingerprint is a print which is obtained by rolling the finger from one side of nail to the other side of nail (namely, nail-to-nail) on a card or inside a platen scanner (shown in Figure 1(a)). The plain is a print collected by pressing the finger down on a card or place the finger flat on a scanner (shown in Figure 1(b)). The latent ones are acquired from crime scenes where the prints are not intentionally touched by the suspects or criminals (shown in Figure 1(c)). For the rolled and plain fingerprints, both are acquired in a controlled mode. That is, they are typically in good quality and are rich of reliable detailed features (e.g. minutiae). Consequently AFIS is able to handle the rolled and plain identification cases in full-automatic mode. In contrast, the fingerprint in latent images are usually small-sized, overlapped with other image components and blurred due to the following possible causes:

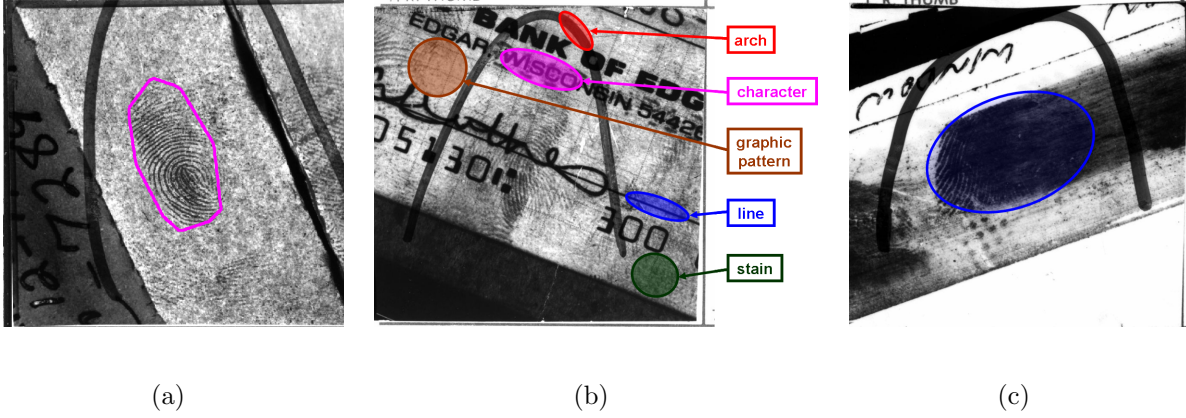


Figure 2: The challenges involved in latent print images: (a) small area in ROI; (b) overlapping with the various structured noise; and (c) blur.

- Small area: the most fingerprints collected from crime scenes are not complete but partial (shown in Figure 2(a));
- Overlapping with other structured components: the fingerprints usually overlap with other structured noise such as arch, line, character, stain, and graphic pattern (shown in Figure 2(b));
- Blur: the most fingerprints acquired from crime scenes have large distortion due to the pressure variations when the fingers touching or pressing down the object surface (shown in Figure 2(c)).

All above adverse effects are not solely encountered but concurrently confronted, therefore latent fingerprint images are generally in poor quality. Considering that AFIS primarily depends on the sufficient and reliable features, the latent fingerprint identification based on automatically extracted features is inaccurate. That is, the poor image quality imposes the difficulties on the automated feature extraction so that the extracted features are limited and the most of them are not reliable. Consequently the fully-automated matching con-

ducted by AFIS based on the limited and unreliable features would lead to the inaccurate result. In order to ensure the reliability and accuracy for latent fingerprint identification, the good-quality ROI as well as the reliable features are often manually marked instead of automatically extracted. With the human intervention involved, for latent fingerprint identification, the semi-automatic mode rather than full-automatic mode is adopted.

The semi-automatic latent fingerprint identification procedure consists of the following four stages: (i) the ROI in latent images are manually labeled; (ii) based on the labeled ROI, the features such as minutiae, singularity, ridge quality map, orientation field, ridge wavelength map, and skeleton are manually extracted [5]; (iii) the marked features are uploaded to a latent fingerprint matcher, then are automatically matched against the features derived from the rolled / plain fingerprints in background database; and (vi) according to the matching scores, the candidate rolled / plain prints are retrieved, and the candidates are visually verified by latent examiners. After visual verification, the most possible archived fingerprint in background database might be found. If not found, a new round routine with more cautious ROI labeling, feature extraction, and visual verification would be conducted for obtaining the most possible candidate. For semi-automatic mode in latent fingerprint identification, the resultant accuracy is satisfactory [6].

In order to reduce the cost of expert work, a fully automated latent fingerprint matching and identification is needed [9]. This is also desirable for the advancement of the technology. [7] and [8] propose a latent matching algorithm without involving many manually marked features and it only used the minutiae information provided by latent experts. Compared with [5] and [6], the most elaborately marked features (e.g. singularity, ridge quality map, orientation field, ridge wavelength map, and skeleton) by human are not considered and

used in the proposed matching system, thus the human intervention is remarkably cut down. However, the minutiae extraction is still proceeded manually. In order to avoid the human intervention and achieve high-degree automation, the development of ROI segmentation method and minutiae set-based latent matcher could be another possible solution. Recently, the ROI detector and the minutiae level matching algorithm have been the subject of several studies.

Several approaches have been proposed to address the problem of ROI segmentation. An automated ROI segmentation technique is presented in [12]. For the proposed technique, however, the local orientation and spatial frequency are estimated by using a local pixel intensity projection which is sensitive to the variation of pixel intensity caused by structured noise. [14] generates a ideal “ridge-valley” pattern template and then uses the cross-correlation between a local image patch and the generated template to evaluate the local fingerprint quality. The frequency of ideal “ridge-valley” pattern template is predefined according to the fixed empirical value, therefore, the generated template is not adaptive to the real local spatial frequency. A total variation (TV) model-based approach is proposed to handle latent fingerprint segmentation task. Therein, the latent fingerprint image is decomposed into cartoon and texture layers and the ROI is detected based on the texture layer by using traditional segmentation methods [11] [10]. However, the proposed TV model incorporates the orientation field which is directly calculated from the original poor-quality latent image. Thus, the directional information used in the proposed TV model is not reliable. [13] presents an automated method based on orientation tensor and local ridge frequency to concurrently localize ROI in latent images. However, the local ridge frequency directly estimated from local Fourier analysis on original latent patch is sensitive to the presence of structured noise.

[9] proposes a dictionary learning-based segmentation and enhancement method, where the multi-layer ridge structure dictionaries from the coarse level to fine level are separately established by using dictionary learning algorithm. Such approach heavily relies on the learned dictionaries and the training patches are pre-selected from the good-quality rolled fingerprint images. It is demonstrated that a dictionary learned from the target image is preferable (target image means the image currently being processed), since such dictionary can be more adaptive to the target image [17]. However, the ridge structure dictionaries are not learned from the query latent images but from the rolled ones. Therefore, the potentially useful “ridge-valley” pattern in latent fingerprint images are not utilized but ignored in [9].

For the reported latent fingerprint matching algorithms, the manual markup of minutiae is regarded as a common practice in latent fingerprint identification cases. This is not only to ensure the matching accuracy, but also to keep the latent matcher in proper working condition. [7] and [8] only consider the manually marked minutiae as the sole input for hough transform (HT)-based matcher. Due to the involvement of manually marked minutiae (ground-truth), the proposed matcher could achieve satisfactory matching result. However, the straightforward adoption of the minutiae extracted by automated computer programs (e.g. Verifinger SDK) on low-quality latent images is most likely to result in the poor matching performance. Because of poor quality and overlapping structured noise in latent images, a fair amount of spurious minutiae are possibly yielded via a full-automatic procedure. Consequently, the proposed latent matcher is not robust but vulnerable to the corruption caused by spurious minutiae. [5] and [6] also propose the base-line matching algorithm which only takes the manually marked minutiae as the matcher input. Similar to [7] and [8], the proposed matching approach is also sensitive to the presence of spurious minutiae. Apparently,

the robustness and tolerance of minutiae-level matcher for spurious minutiae is therefore an important property and plays a critical role when fulfilling the poor-quality fingerprint matching duties.

The work on robust fingerprint matcher for rolled prints has been proposed. [15] proposes a fingerprint matcher based on genetic algorithm (GA) in order to deal with the significant occlusion and clutter of minutiae caused by low-quality prints. This method achieves good performance when handling low-quality rolled prints. However, the matching performance for latent prints is still unknown. Further, the proposed matcher heavily depends on the local minutiae triangle-based fitness function. Since the local triangle generation based on each triplet of minutiae is computationally intensive, such type of fitness function is too inefficient for GA-based optimization to solve the large size minutiae set matching problem. In this paper, we propose a robust minutiae set-based matcher embedding with a self-learning module for ROI identification in latent fingerprint images. The proposed latent matcher integrates the following two modules: (i) the dictionary learning (DL)-based ROI segmentation scheme; and (ii) the GA-based minutiae set matching unit. For the DL-based ROI segmentation scheme, the dictionary is firstly learned from the query latent fingerprint image. Then, based on the learned dictionary, the “ridge-valley” pattern elements (dictionary atoms) can be automatically identified. Further, the sparse representation for the original latent image patches is performed. Finally, depending on the presence or absence of the sparse coefficients that are corresponding to the identified “ridge-valley” atoms, the foreground (fingerprint region) is segmented. In the GA-based minutiae-level matching unit, the two minutiae sets, one from the segmented ROI in query latent image (obtained via segmentation module) and the other one from the print currently being compared, are beforehand extracted through a



normal automated minutiae extraction program like Verifinger SDK which is widely available to the public. Then, according to the affine transformation parameters estimated by GA, the minutiae set alignment between the query latent and the compared print is performed. Further, after aligning two sets of minutiae, the correspondence between the two sets needs to be found. Accordingly, the corresponding minutiae points between the query latent and the compared print could be paired. Finally, the number of matched minutiae is obtained and simply regarded as the matching score.

The main contributions of this paper are summarized as follows:

1. To our knowledge, there is no state-of-art latent fingerprint matcher in available public domain. Therefore, in this paper, we introduce a multi-module matcher to cope with the latent fingerprint matching problem. The proposed system is performed in a full-automatic mode. Experimental results based on NIST SD27 demonstrate that the proposed matcher with the proposed segmentation module (SM) (say, proposed matcher + proposed SM) can achieve 34.496% penetration rate. The comparative experiments have been conducted to evaluate the effect of the different SMs by using the proposed matcher with and without SM. By designating the proposed matcher without SM as the baseline (say, proposed matcher only), such benchmark penetration rate is 38.159%. Based on the benchmark, the proposed matcher with the state-of-art SM such as [9] (say, proposed matcher + SM [9]) only achieves 36.434% penetration rate. The relative penetration rate enhancement percentage for “proposed matcher + proposed SM” ( $9.59\% = \frac{|34.496\% - 38.159\%|}{38.159\%}$ ) is at least twice better than that of “proposed matcher + SM [9]” ( $4.52\% = \frac{|36.434\% - 38.159\%|}{38.159\%}$ ).

2. The fully automated ROI segmentation module is plug into the proposed latent matcher and is performed as the pre-processing for the subsequent matching task. The proposed SM consists of the following phases: (i) the image structure dictionary learning; (ii) the “ridge-valley” atom identification; and (iii) the sparse coding and ROI segmentation. Existing method requires to establish a dictionary from the high-quality rolled image patches in advance [9]. Different from such conventional methods, we propose to build up the structure dictionary directly learned from the query latent image. As demonstrated in [17], a learned dictionary based on the target image can better adapt to the target image. Therefore, the dictionary obtained in the proposed SM is not good quality rolled image patches-determined but query latent image-oriented.
  
3. The robust latent matching unit consists of the following stages: (i) the ROI-based minutiae extraction; (ii) the GA-based minutiae set alignment; and (iii) the counting of paired minutiae. Existing method demands to yield the local minutiae descriptors during the iteration of GA optimization [15]. Different from such conventional method, the global topology of the entire minutiae set is directly adopted in the proposed matching unit instead of the local minutiae structure. Accordingly, the proposed matching unit is more robust to the presence of spurious minutiae and more efficient in the matching.

The rest of this paper is organized as follows: in Section II, the details of the proposed matching system are introduced; in Section III, ROI-based minutiae extraction, and latent fingerprint matching experiments are implemented respectively. In the ROI-based minutiae extraction experiment, by adopting the obtained ROI in automated segmentation module,

the reduction of spurious minutiae points as well as the preservation of genuine ones are assessed. In latent fingerprint matching experiment, the matching performance of the introduced latent matcher is evaluated; in Section IV, the conclusions and on-going research directions are presented.

## 2 Proposed Latent Fingerprint Matching System

In this section, the proposed multi-module matching system for latent fingerprint is proposed, which consists of the two following modules: (i) the dictionary learning-based ROI segmentation scheme; and (ii) the genetic algorithm (GA)-based minutiae set matching unit.

### 2.1 Dictionary Learning-Based ROI Segmentation Module

#### 2.1.1 Query Latent Image-Based Dictionary Learning

The dictionary learning procedure is performed based on the query latent fingerprint image instead of the good-quality rolled or plain print images. As suggested in [17], a learned dictionary based on the target image can better adapt to the target image and more specifically represent the intrinsic signal structure in target image. Compared with other images-based learned dictionary, the target image-based dictionary can more effectively keep the scale consistency for the real signal structures. That is, the signal structure scales in other images might be more or less different from the ones in target image. Therefore, the other images-based dictionary atoms often do not fittingly model the real-scale structures in target image. Accordingly, the image restoration tasks such as denoising and inpainting are conducted depending on the target image-based dictionary rather than the other image-based dictionary

or the pre-defined analytic dictionary (e.g. wavelets, curvelets and DCT) [17]. Inspired by the advantage of the target image-based dictionary, the dictionary learned from the query latent image is beneficial to the following “ridge-valley” atom identification phase.

The dictionary learning for query latent image can be mathematically formulated as follows: let  $S = \{s_i | i = 1, 2, \dots, N\}$  as the training set, where  $s_i$  is the vector obtained after the vectorization of latent image block  $p_i$  with size  $w \times w$  and  $N$  is the number of selected image blocks in latent fingerprint image. The purpose of dictionary learning is to establish a numerical dictionary  $D$  with size  $N_s \times N_a$  based on the training set  $S$  ( $N_s$  denotes the atom vector dimension where  $N_s = w \times w$  and  $N_a$  stands for the atom number). Such established dictionary  $D$  can effectively represent the vectorized image block  $s_i$  in a sparse way. In order to obtain dictionary  $D$ , the following optimization problem needs to be solved

$$\min_{D, \Gamma} \|S - D\Gamma\|_F^2 \quad s.t. \quad \forall i, \quad \|\gamma_i\|_0 \leq K \quad (1)$$

where  $\Gamma$  is the sparse coefficient matrix with size  $N_a \times N$ .  $\gamma_i$  is the  $i^{th}$  column vector in sparse coefficient matrix  $\Gamma$  and is also corresponding to the  $i^{th}$  vectorized image block  $s_i$ .  $\|\cdot\|_F$  and  $\|\cdot\|_0$  denote the Frobenius norm and  $l_0$  norm respectively.  $\|\gamma_i\|_0$  is equal to the number of nonzero elements in vector  $\gamma_i$ . Therefore, in order to obtain the sparse vector  $\gamma_i$ , the regulator  $\|\gamma_i\|_0 \leq K$  controls that the nonzero elements in vector  $\gamma_i$  should be less than the sparsity parameter  $K$ . Although  $l_0$  is the limit of  $l_p$  when  $p$  approaches zero, it is not a true norm (unlike the  $l_1$  norm which has all properties of a true norm) and also leads to the NP-hard problem. In order to avoid the NP-hard problem, the above  $l_0$  norm-based regulator  $\|\gamma_i\|_0 \leq K$  in Equation (1) can be replaced by the convex  $l_1$  norm-based regulator

$\min(\|\gamma_i\|_1)$ . Accordingly, Equation (1) is updated by the following approximate form

$$\min_{D, \Gamma} \left( \|S - D\Gamma\|_2^2 + \lambda \sum_{i=1}^N \|\gamma_i\|_1 \right) \quad (2)$$

where Frobenius norm  $\|\cdot\|_F$  has been embodied by  $l_2$  norm and  $\lambda$  is the Lagrange coefficient to balance the data fitting and sparsity level. For solving the optimization problem in Equation (2), the sparse coefficient matrix  $\Gamma$  and the dictionary  $D$  are alternatively updated as follows: (i) keeping  $D$  fixed, compute the sparse coefficient matrix  $\Gamma$ ; then (ii) keeping  $\Gamma$  fixed, update the dictionary  $D$ . The above two steps are iteratively repeated until the convergence. In order to learn a dictionary  $D$ , one can apply any available dictionary learning technique such as MOD [16], KSVD [17] and ODL [19]. As reported in [19], an alternate optimization of sparse coding and dictionary update is performed based on a subset of the training data. Such subset continues to be augmented with a new coming training sample. Based on the outcome of the previous iteration, the same alternate optimization is executed again for the new coming training data. ODL iteratively repeats until all training samples have been used. As demonstrated by the experiments in [19], ODL is more faster than MOD and KSVD. Because the large size of training set is produced based on the dense training set pick-up strategy and the whole training set has to be used by MOD and KSVD at each iteration, both MOD and KSVD are computationally expensive. Therefore, instead of MOD and KSVD, ODL is applied to learn the dictionary for query latent fingerprint image due to its more efficient learning mechanism and lower computational complexity. The dictionary  $D$  learned from the given query latent image by using ODL is shown in Figure 3.

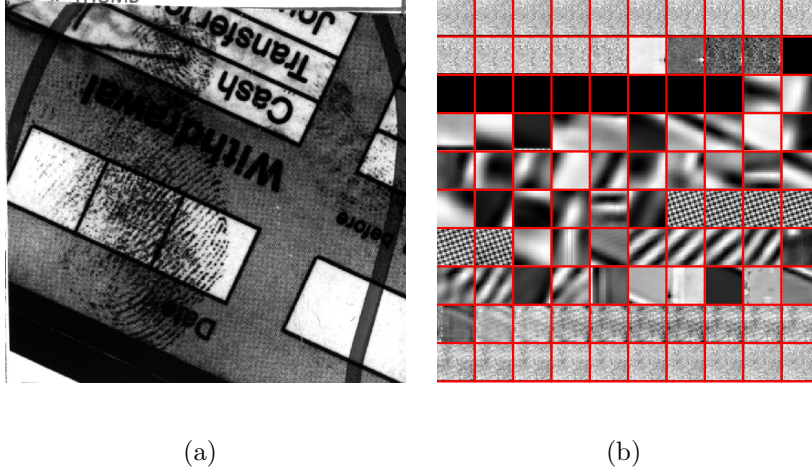


Figure 3: The dictionary  $D$  is learned directly from the given query latent image by ODL: (a) original query latent image (U224 in NIST SD27); and (b) its learned dictionary  $D$  ( $N_a = 100$  and  $K = 2$ ).

### 2.1.2 Ridge-Valley Atom Identification

The atoms with “ridge-valley” pattern need to be distinguished from the ones which are not structured as “ridge-valley” pattern. The detection or modeling of “ridge-valley” pattern is not a new subject and the previous works have been reported in [12] and [14], however, it is still a challenging task. Although “ridge-valley” pattern is one type of intrinsic signal structure in latent fingerprint image, such pattern is usually mingled with other types of structured noise (e.g. arch, line, character, stain, speckle and motif) and even vague or disconnected due to the wet or dry prints. Accordingly, the direct detection or straightforward modeling for such pattern based on original latent image patch is inaccurate. In contrast to the original patch-level detection, the atom-level detection is easier and more reliable. That is, the dictionary learning procedure is not only for the signal structural decomposition but also for the signal structure refinement. After the completion of dictionary learning,

the intrinsic signal structures at atom level becomes more salient and easier to be distinguished. Because the identified “ridge-valley” atoms play a crucial role in the subsequent sparse coefficient-based ROI segmentation phase, the development of a fully automated identification approach is necessary.

In this study, a local Fourier analysis and cross correlation-based method is proposed, which consists of the following steps:

- *Step 1:* Given a learned dictionary  $D = \{d_k | k = 1, 2, \dots, N_a\}$ , each single atom vector  $d_k$  with dimension  $N_s$  needs to be converted to the atom patch  $p_{d_k}$  with size  $w \times w$  ( $N_s = w \times w$ ) (shown in Figure 4(a));
- *Step 2:* Input a atom patch  $p_{d_k}$ , the 2D Discrete Fourier Transform (2D DFT) is applied to  $p_{d_k}$ , then the spectrum  $DFT(p_{d_k})$  in frequency domain is obtained (shown in Figure 4(b));
- *Step 3:* In the spectral magnitude map  $|DFT(p_{d_k})|$  (rearranged by moving the zero-frequency component to the center), the paired highest magnitude points are detected within the frequency range corresponding to the ridge period range  $[3 \text{ pixels}, 20 \text{ pixels}]$ . Such ridge period range can cover most image underlying structures from low spatial frequency to high spatial frequency. The bandpass constraint in frequency domain corresponding to the ridge period range  $[3 \text{ pixels}, 20 \text{ pixels}]$  in spatial domain is shown in Figure 4(c). Such frequency bandpass filter masks over the spectral magnitude map to yield the constrained spectral magnitude map, and the highest magnitude points can be detected and shown in Figure 4(d) (marked by red squares).

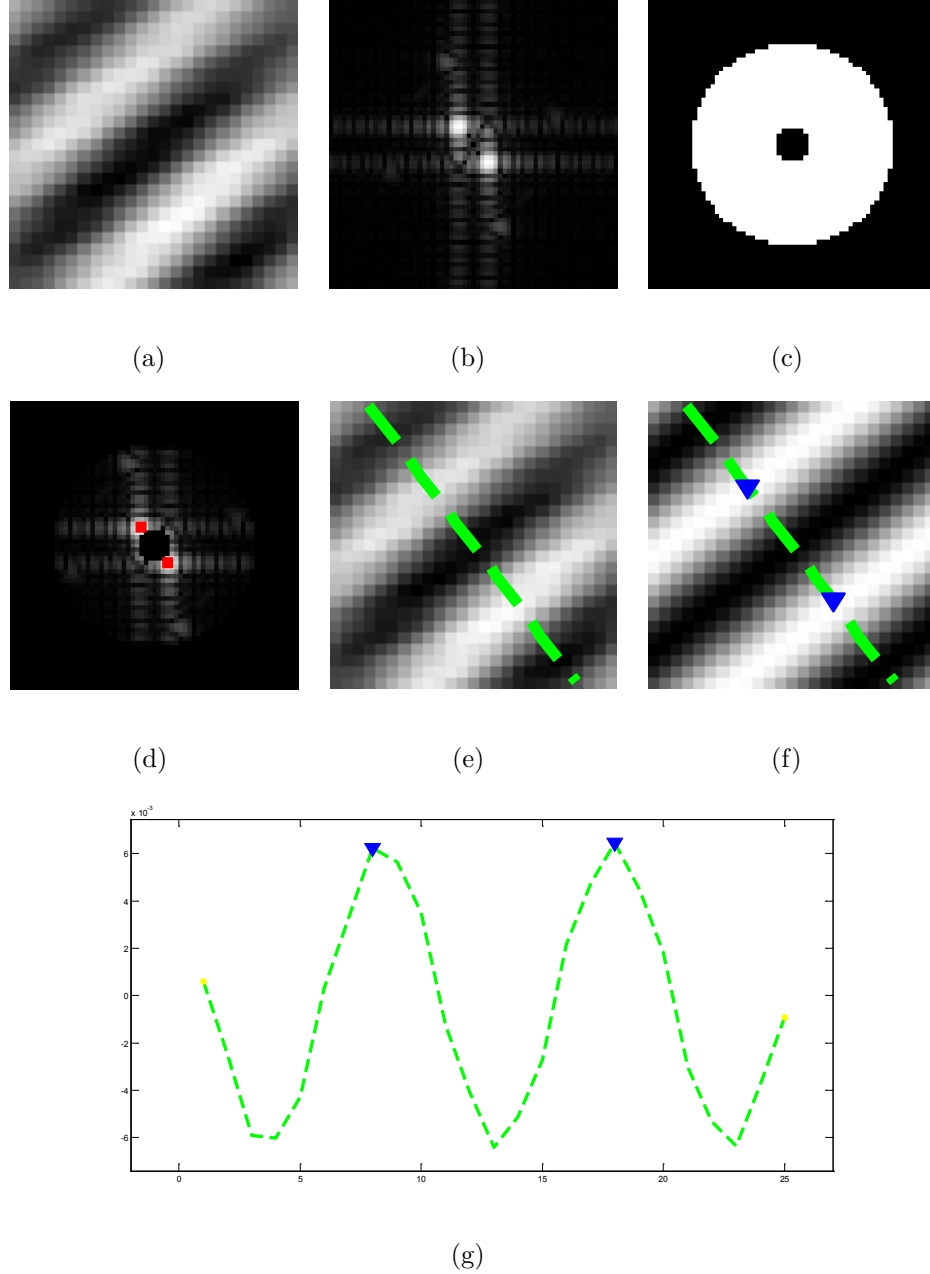
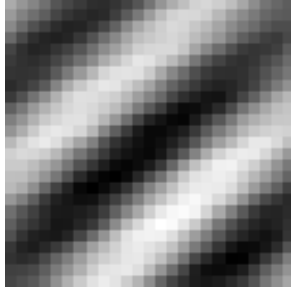
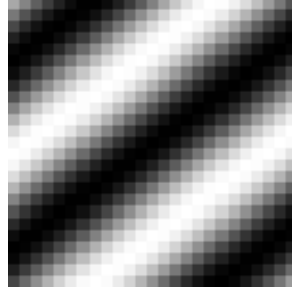


Figure 4: The illustration of the details during the automated “ridge-valley” atom identification procedure: (a) the “ridge-valley” atom patch ( $p_{d_{66}}$  in Figure 3(b)); (b) the spectral magnitude map  $|DFT(p_{d_{66}})|$ ; (c) the frequency bandpass filter corresponding to the ridge period range  $[3 \text{ pixels}, 20 \text{ pixels}]$ ; (d) the constrained spectral magnitude map depending on (c); (e) the calculated orientation  $o_{d_{66}}$ ; (f) the reconstructed “ridge-valley” pattern  $u_{d_{66}}$ ; and (g) the 1D sinusoidal-shaped wave modeled based on the pixel intensities along the green dashed line indicated in (f).





(a)

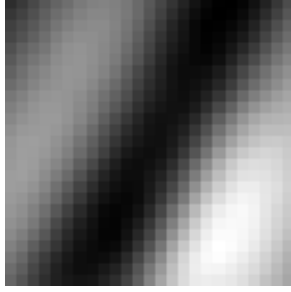


(b)

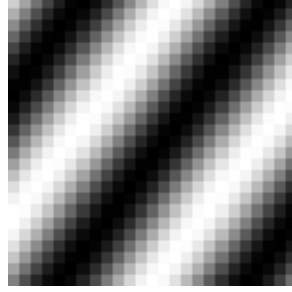
$$xcorr_{p_{d_{66}}, u_{d_{66}}} = 0.9512$$

$$v_{d_{66}} = 10 \text{ pixels}$$

(c)



(d)

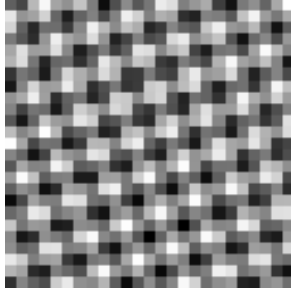


(e)

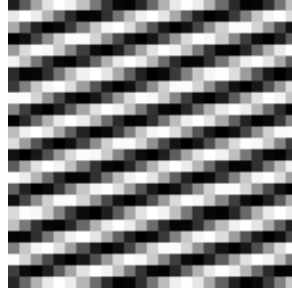
$$xcorr_{p_{d_{45}}, u_{d_{45}}} = 0.4257$$

$$v_{d_{45}} = 10 \text{ pixels}$$

(f)



(g)



(h)

$$xcorr_{p_{d_{57}}, u_{d_{57}}} = 0.6819$$

$$v_{d_{57}} = 3.1667 \text{ pixels}$$

(i)

Figure 5: The examples of atom-based identification for various atoms: (a) the “ridge-valley” atom ( $p_{d_{66}}$  in Figure 3(b)); (b) and (c) are the reconstructed  $u_{d_{66}}$ , the calculated  $xcorr_{p_{d_{66}}, u_{d_{66}}}$  and  $v_{d_{66}}$  for (a) respectively; (d) the line-like atom ( $p_{d_{45}}$  in Figure 3(b)); (e) and (f) are the reconstructed  $u_{d_{45}}$ , the calculated  $xcorr_{p_{d_{45}}, u_{d_{45}}}$  and  $v_{d_{45}}$  for (d) respectively; (g) the periodical speckle atom ( $p_{d_{57}}$  in Figure 3(b)); (h) and (i) are the reconstructed  $u_{d_{57}}$ , the calculated  $xcorr_{p_{d_{57}}, u_{d_{57}}}$  and  $v_{d_{57}}$  for (g) respectively.

- *Step 4:* Based on the coordinate of paired points detected in *Step 3*, the orientation  $o_{d_k}$  for the currently processed atom patch  $p_{d_k}$  is calculated (for “ridge-valley” atom, the orientation  $o_{d_k}$  is not along but across the ridge, as indicated by the green dashed line in Figure 4(e));
- *Step 5:* Reconstruct the “ridge-valley” pattern  $u_{d_k}$  according to the magnitude and phase corresponding to the detected points in constrained spectral magnitude map (the reconstructed  $u_{d_k}$  is illustrated in Figure 4(f)), then calculate the cross-correlation value  $xcorr_{p_{d_k}, u_{d_k}}$  between the atom patch  $p_{d_k}$  and the reconstructed “ridge-valley” pattern  $u_{d_k}$  according to Equation (3).

$$xcorr_{p_{d_k}, u_{d_k}} = \frac{\sum_{x,y} (\alpha \cdot \beta)}{\sqrt{\sum_{x,y} (\alpha^2) \cdot \sum_{x,y} (\beta^2)}} \quad (3)$$

where  $\alpha = f_{p_{d_k}}(x, y) - \bar{f}_{p_{d_k}, u_{d_k}}$  and  $\beta = f_{u_{d_k}}(x - a, y - b) - \bar{f}_{u_{d_k}}$ .  $f_{p_{d_k}}(x, y)$  and  $f_{u_{d_k}}(x, y)$  denote the atom image  $p_{d_k}$  and the reconstructed pattern  $u_{d_k}$  respectively.  $\bar{f}_{p_{d_k}, u_{d_k}}$  stands for the mean of  $f_{p_{d_k}}(x, y)$  when overlapping with  $f_{u_{d_k}}(x, y)$ .  $\bar{f}_{u_{d_k}}$  is the mean of the entire reconstructed pattern image  $u_{d_k}$ .  $a$  and  $b$  denote the offsets along  $x$ - and  $y$ - axis respectively. Equation (3) indicates the pattern similarity between atom patch  $p_{d_k}$  and “ridge-valley” pattern  $u_{d_k}$ . The higher cross-correlation value is, the atom patch  $p_{d_k}$  is more likely to represent the “ridge-valley” pattern.

- *Step 6:* For the reconstructed “ridge-valley” pattern  $u_{d_k}$ , the pixel intensities along the orientation  $o_{d_k}$  are acquired to model a 1D sinusoidal-shaped wave, then the wave peaks are detected to calculate the spatial ridge period  $v_{d_k}$ . The pixels along  $o_{d_k}$  are

highlighted by the green dashed line in Figure 4(f), and the modeled 1D sinusoidal-shaped wave is shown in Figure 4(g) where the blue triangles indicate the detected wave peaks;

- *Step 7:* Check whether the following two criterions are concurrently satisfied: (i)  $xcorr_{p_{d_k}, u_{d_k}} \geq Th_{xcorr}$  ( $Th_{xcorr}$  is the threshold to determine the pattern similarity); and (ii)  $v_{d_k} \in [5.3 \text{ pixels}, 12.8 \text{ pixels}]$  ( $[5.3 \text{ pixels}, 12.8 \text{ pixels}]$  is suggested in [13]. In the proposed atom identification procedure, the broad range  $[3 \text{ pixels}, 20 \text{ pixels}]$  is firstly adopted to involve more candidate atoms, then the narrow range  $[5.3 \text{ pixels}, 12.8 \text{ pixels}]$  is applied to filter out the unqualified atoms). If both satisfied, the currently processed atom patch  $p_{d_k}$  is regarded as the “ridge-valley” atom. Otherwise, label  $p_{d_k}$  as the “non ridge-valley” atom. The judgement depending on the single criterion is too limited to make a correct decision on atom level, therefore both criterions need to be simultaneously satisfied. As demonstrated in Figure 5, the “non ridge-valley” atoms either satisfies  $xcorr_{p_{d_k}, u_{d_k}} \geq Th_{xcorr}$  (here  $Th_{xcorr} = 0.6$ ) or meets  $v_{d_k} \in [5.3 \text{ pixels}, 12.8 \text{ pixels}]$ , however, they are not the “ridge-valley” atoms indeed;
- *Step 8:* Check whether all the atoms in  $D$  have been judged. If not, go back to *Step 2*. Otherwise, terminate the “ridge-valley” atom identification procedure and output all the atom labels. An example of the “ridge-valley” atoms identification for all the atoms in learned dictionary is illustrated in Figure 6.

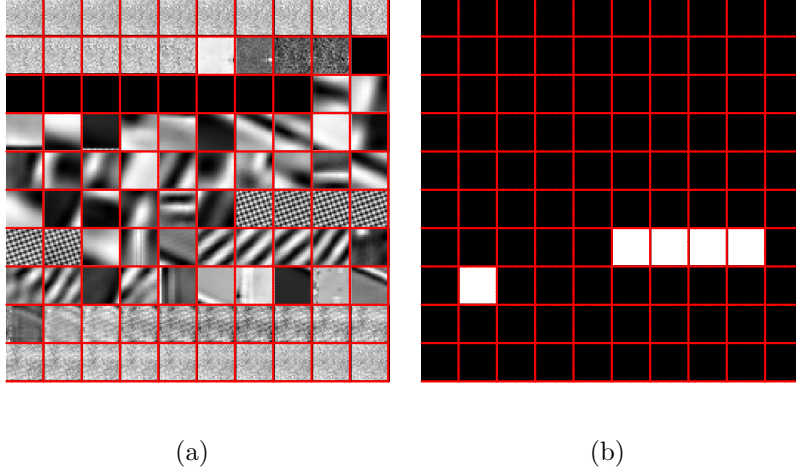


Figure 6: The example of the identification results obtained by using the proposed automated “ridge-valley” atom identification procedure: (a) the learned dictionary  $D$  in Figure 3(b); and (b) the identified “ridge-valley” atoms. The white patches indicate that the original atoms with the same patch-wise coordinates in (a) are “ridge-valley” ones, while the black patches label that the corresponding atoms in (a) are “non-ridge-valley” ones.

### 2.1.3 Sparse Coefficient-Based ROI Segmentation

The sparse coefficients by projecting the image blocks in query latent image onto the learned dictionary are utilized to determine whether the original image blocks belong to the foreground. To be specific, the sparse projection from every single latent image block to the learned dictionary yields the sparse coefficient vector. Given a single latent image block, the elements inside its sparse coefficient vector are quite different. That is, the most elements are zero while few are nonzero. The nonzero elements in sparse coefficient vector measures the similarity between such latent image block and the corresponding atoms. Among the nonzero elements, the higher one indicates the corresponding atom is more similar to such image block, while the lower one indicates the lower similarity. As a latent image block with

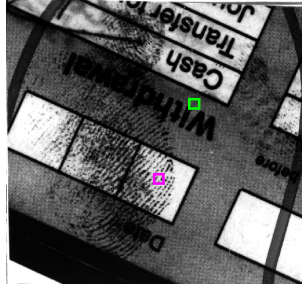
“ridge-valley” pattern is considered as one part of real fingerprint, the latent image block whose highest nonzero element in sparse coefficient vector corresponds to the “ridge-valley” atom is regarded as the subset of foreground. Otherwise, the image block whose highest nonzero element in sparse coefficient vector corresponds to the “non-ridge-valley” atom is regarded as the background. Motivated by the presence or absence of highest nonzero sparse coefficients corresponding to the identified “ridge-valley” atoms in preceding stage, the ROI segmentation phase could be smoothly proceeded.

The generation of sparse coefficients can be mathematically formulated as the following equation

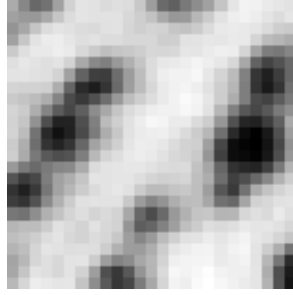
$$\tilde{s}_i = \gamma_i^{(1)} d_1 + \gamma_i^{(2)} d_2 + \gamma_i^{(3)} d_3 + \dots + \gamma_i^{(N_a)} d_{N_a} = \sum_{k=1}^{N_a} \gamma_i^{(k)} d_k \quad (4)$$

where  $\tilde{s}_i$  is the approximation of the given signal vector  $s_i$  ( $s_i$  is obtained after the vectorization for the latent image block  $p_i$ ).  $\gamma_i^{(k)}$  is the element inside the sparse coefficient vector  $\gamma_i = [\gamma_i^{(1)}, \gamma_i^{(2)}, \gamma_i^{(3)}, \dots, \gamma_i^{(k)}, \dots, \gamma_i^{(N_a)}]^T$ . Due to the sparsity constraint in Equation (1), most atoms in dictionary  $D$  are not selected for the representation of  $s_i$ , but only a small number of atoms are adopted. Accordingly, most elements in vector  $\gamma_i$  are zero while few are nonzero. In this study, the nonzero elements inside the sparse coefficient vector  $\gamma_i$  are used for ROI segmentation. Such sparse coefficient-based ROI segmentation stage consists of the following steps:

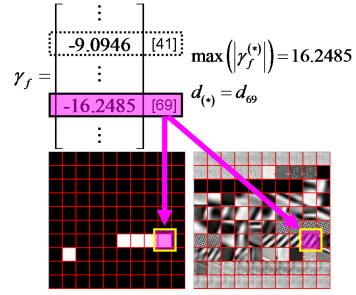
- *Step 1:* Initialize a vacant image  $M$  (all the pixel intensities are zero) with the same size as the original latent fingerprint image, then divide the latent image into the overlapping blocks (block size  $w \times w$ , as shown in Figure 7(b) and Figure 7(d)) and



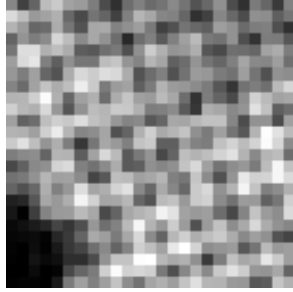
(a)



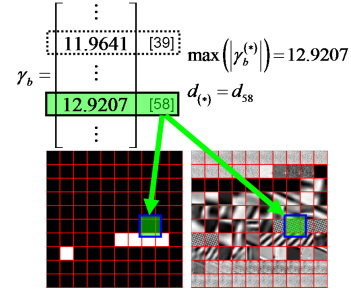
(b)



(c)



(d)



(e)

Figure 7: The illustration of the details during the automated sparse coefficient-based ROI segmentation procedure: (a) the two types of image blocks in latent print image - the foreground block (marked by pink square) and the background block (marked by green square); (b) the marked foreground block; (c) seeking for the specific atom corresponding to the  $\max(|\gamma_f^{(*)}|)$  for (b) (the found atom is identified as the “ridge-valley” atom in preceding phase); (d) the marked background block; and (e) seeking for the specific atom corresponding to the  $\max(|\gamma_b^{(*)}|)$  for (d) (the found atom is labeled as the “non-ridge-valley” atom in preceding phase).

rearrange the obtained image blocks into the column vectors (here the training set  $S = \{s_i | i = 1, 2, \dots, N\}$  for dictionary learning is directly used, where  $s_i$  is obtained after the vectorization for the image block  $p_i$ );

- *Step 2:* Given a signal vector  $s_i$ , orthogonal matching pursuit (OMP) [20] is applied to the given vector  $s_i$  for selecting few atoms to sparsely approximate  $s_i$ , then the sparse coefficient vector  $\gamma_i$  is obtained;
- *Step 3:* Find the highest nonzero absolute value  $|\gamma_i^{(*)}|$  in sparse coefficient vector  $\gamma_i$  and the corresponding atom  $d_{(*)}$  in dictionary  $D$  (shown in Figure 7(c) and Figure 7(e));
- *Step 4:* Check whether the found atom  $d_{(*)}$  is the identified “ridge-valley” atom (shown in Figure 7(c) and Figure 7(e)). Further, the pixel intensities inside the block of  $M$  whose size and block-wise coordinate are the same as the currently processed block  $p_i$  in latent image, are tuned according to Equation (5).

$$M_{p_i}(x, y) = \begin{cases} M_{p_i}(x, y) + 1, & d_{(*)} \text{ is "ridge - valley" atom} \\ M_{p_i}(x, y), & d_{(*)} \text{ is not "ridge - valley" atom} \end{cases} \quad (5)$$

where  $M_{p_i}(x, y)$  denotes the pixel intensities inside the block of  $M$ , whose size and block-wise coordinate are the same as the currently processed block  $p_i$  in latent image.

- *Step 5:* Check whether all the signal vectors have been processed. If not, go back to *Step 2*. Otherwise, terminate the calculation of sparse coefficient for signal vector and output the image  $M$  (shown in Figure 8(a));

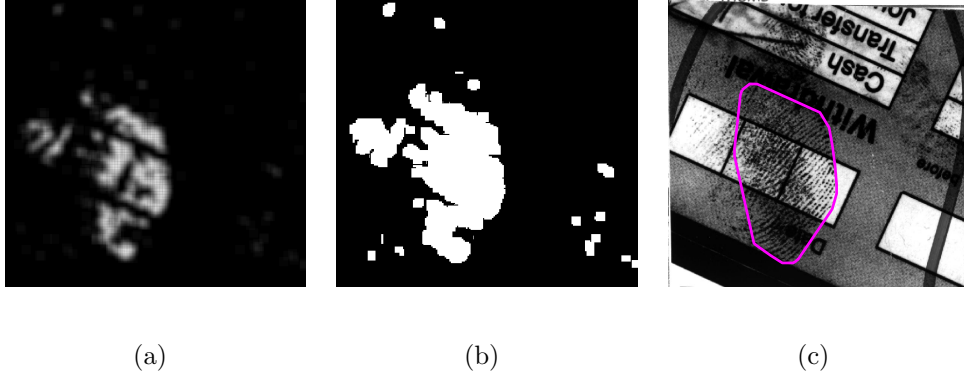


Figure 8: The example of the ROI obtained by using the proposed sparse coefficient-based ROI segmentation procedure: (a) the image  $M$  obtained after all the signal vectors have been processed; (b) the binary image obtained by using the normalization and Otsu’s adaptive threshold-based segmentation method; and (c) the polygonal ROI obtained via the mathematical morphology operation and the maximal-area convex polygon extraction (marked by pink convex polygon over the original latent print image U224 in NIST SD27).

- *Step 6:* All the pixel intensities in image  $M$  need to be normalized to the range  $[0, 1]$ , then the normalized image is binarized by Otsu’s adaptive threshold-based segmentation method [25] (shown in Figure 8(b));
- *Step 7:* A set of mathematical morphology operators such as close, open, hole-filling, and small block removal are applied to the obtained binary image, finally the maximal-area convex polygon containing the foreground regions is achieved as the ROI (shown in Figure 8(c)).



## 2.2 Genetic Algorithm-Based Latent Matching Unit

### 2.2.1 ROI-Based Minutiae Extraction

The minutiae sets for latent fingerprint images are extracted from the ROI obtained in segmentation module. As demonstrated in [7] [8] [5] and [6], the reliability of minutiae plays a crucial role in the performance of latent matcher. Obviously, the involvement of spurious minutiae deteriorates the matching performance. That is, the more spurious involved, the poorer the matcher behaves. However, how to effectively reduce or even avoid the spurious minutiae in fully-automatic mode becomes a significant issue. As the outcome of segmentation scheme, the resultant ROI is used for automated minutiae extraction which can not only effectively preserve the genuine minutiae but also significantly reduce the false ones. That is, the genuine minutiae are most likely to be detected within ROI while the spurious ones caused by the structured noise in background are effectively eliminated by ROI mask. Therefore, the minutiae set obtained from ROI in latent image is supposed to be reliable.

### 2.2.2 Problem Formulation for Minutiae Set-Based Fingerprint Matching

The minutiae set-based fingerprint matching can be mathematically formulated as follows: let  $C = \{c_i | i = 1, 2, \dots, m\}$  and  $L = \{l_j | j = 1, 2, \dots, n\}$  be the two point set in  $R^2$  space ( $C$  stands for the minutiae set extracted from the currently compared print and  $L$  denotes the minutiae set extracted from the ROI in query latent print respectively). Both sets determine whether there is a specific affine transformation  $T = (\theta, s, t_x, t_y)$  ( $\theta$  denotes the rotation angle,  $s$  represents the scaling factor, and  $t_x$  and  $t_y$  are the offsets along  $x$ - and  $y$ - axis respectively) that maps set  $C$  onto or close to set  $L$  to indicate a correspondence. Therefore,

seeking for the affine transformation as well as the correspondences (an exact one-to-one correspondence, or an approximate correspondence) between both point sets  $C$  and  $L$  are coupled point matching problem. Here let  $(c_i, l_j)$  be one of the corresponding pairs under  $T$ , and denote  $c_i = (x_{c_i}, y_{c_i}, o_{c_i}, pt_{c_i})^T$  and  $l_j = (x_{l_j}, y_{l_j}, o_{l_j}, pt_{l_j})^T$ .  $(x_{c_i}, y_{c_i})$  and  $(x_{l_j}, y_{l_j})$  are corresponding coordinates,  $o_{c_i}$ ,  $o_{l_j}$ ,  $pt_{c_i}$  and  $pt_{l_j}$  denote the point orientation and type of  $c_i$  and  $l_j$  respectively. The formula of affine transformation is denoted as follows

$$(x_{l_j}, y_{l_j}) = T[(x_{c_i}, y_{c_i})] \quad (6)$$

$$\begin{bmatrix} x_{l_j} \\ y_{l_j} \end{bmatrix} = s \cdot \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \cdot \begin{bmatrix} x_{c_i} \\ y_{c_i} \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \quad (7)$$

Equation (7) indicates that the coupled point matching problem can be regarded as the parameter tuning problem. That is, with the optimal selection for the parameter, the corresponding affine transformation is generated. Given the optimal affine transformation, set  $C$  are mapped onto set  $L$ . After transformation, the most points in set  $C$  being close to the points in set  $L$  are regarded as matched points.

### 2.2.3 Minutiae Set-Based Matching Problem Solved by GA

Seeking for the appropriate affine transformation parameters is important for the alignment between latent minutiae set and the compared print minutiae set. Such suitable parameters ensure the largest overlap of global topology between two point sets. Considering that the affine transformation parameters need to be elaborately tuned, thus the parameter optimization technique is necessary. In contrast to the deterministic optimization algorithm like

greedy searching method proposed in [5] and [6], the evolutionary optimization approaches have the advantage of computational efficiency and the capability to effectively avoid the local optimum. As a consequence, GA, one of the typical evolutionary optimization approaches, is applied to solve the minutiae set-based matching problem.

To be specific, GA begins with the randomly initialized chromosomes which represent the solution of problem. In subsequent iteration, the updated chromosomes are obtained by using various genetic operators. According to the fitness function, the previous chromosomes are substituted by the updated ones when the updated ones are judged to be fitter individuals than the previous ones. With the continuous iteration, the chromosomes are motivated to evolve to the fittest individuals until the termination of algorithm. Because GA is not easy to trap into local optimal and its searching manner is potentially parallel, it has been broadly utilized to solve the point set matching problem [22] [23] and [24].

In order to use GA, the affine transformation parameters, namely rotation angle  $\theta$ , scaling factor  $s$ , and translation offsets  $t_x$  and  $t_y$  are coded as chromosome. The chromosome vector is denoted as follow

$$Chrom = (\theta, s, t_x, t_y)^T \quad (8)$$

where each parameter value is a random real number and is restricted in an appropriate range. Compared with the binary coding, the real number-based coding has the following advantages: effectively avert the hamming cliff and avoid the decimal digits assignment. Each encoded chromosome corresponds to one parameter set of affine transformation to align the point set. During the iterations of GA, the chromosomes are constrained in the

same range as they are randomly initialized.

To motivate the evolution of GA, the fitness of each chromosome, needs to be evaluated. In order to define the fitness function, coordinate distance  $e_d$  and point orientation difference  $e_o$  need to be calculated in advance

$$e_d = \sqrt{[T(x_{c_i}) - x_{l_j}]^2 + [T(y_{c_i}) - y_{l_j}]^2} \quad (9)$$

$$e_o = |T(o_{c_i}) - o_{l_j}| \quad (10)$$

where  $T(o_{c_i})$  is point  $c_i$ 's orientation after rotation caused by transformation  $T$ . Given the resultant  $e_d$  and  $e_o$ , the fitness function is defined as follow

$$num^t = \begin{cases} num^{t-1} + 1, & \text{if } e_d \leq \delta_d, e_o \leq \delta_o, pt_{c_i} = pt_{l_j} \\ num^{t-1}, & \text{otherwise} \end{cases} \quad (11)$$

where  $num^t$  and  $num^{t-1}$  are the numbers of matched point pairs in  $t^{th}$  and  $(t-1)^{th}$  iteration respectively. Therein, the number of paired points is directly assigned as the fitness function value.  $\delta_d$  and  $\delta_o$  are the tolerances for  $e_d$  and  $e_o$  respectively. Equation (11) indicates that for every iteration all the matched point pairs need to be found and counted when the following three criterions are simultaneously satisfied: (i)  $e_d \leq \delta_d$ ; (ii)  $e_o \leq \delta_o$ ; and (iii)  $pt_{c_i} = pt_{l_j}$ . Therefore, a chromosome producing larger  $num$  (higher fitness) is considered to be superior to the other chromosomes with smaller  $num$  (lower fitness). As such, the fittest chromosomes can be determined according to the fitness values of all chromosomes. In addition, Equations (9-11) not only consider the global topology of point set but also involve the point property

such as point orientation and type during the iterative evolution of GA.

To boost the evolution of GA, the GA operators such as selection, crossover and mutation are also important. To be more specific, the selection operator is used to select the chromosomes with higher fitness, which preserve and inherit the information of these fitter chromosomes into next generation. Therein, the widely used Roulette Wheel selection scheme is applied and the selection probability of each chromosome is proportional to its own fitness value. Besides, the creation of new offsprings to enhance the diversity of chromosomes is necessary because only the inheritance of fitter chromosomes in iterative evolution is not sufficient. Therefore, crossover operator produces new chromosomes through combining partial segments of two parent chromosomes. Therein, the multi-point crossover operator is adopted. Moreover, in order to further enhance chromosomes diversity, the uniform mutation operator is performed to randomly shift the value of chromosome vector with a small probability.

In this study, the GA-based minutiae set matching algorithm is summarized as follows:

- *Step 1:* Extract the minutiae set  $C$  from the currently compared print and the minutiae set  $L$  from the ROI in query latent print (shown in Figure 9(a)), then set population size of chromosomes  $S_{chrom}$ , crossover probability  $p_c$ , mutation probability  $p_m$ , the maximal iterations  $g_{max}$ , point coordinate distance tolerance  $\delta_d$ , point orientation difference tolerance  $\delta_o$  and the value range for chromosome vector;
- *Step 2:* Randomly generate chromosomes within the value range as initial generation;
- *Step 3:* Compute fitness values for all chromosomes in current generation and then select the fitter ones by selection operator;

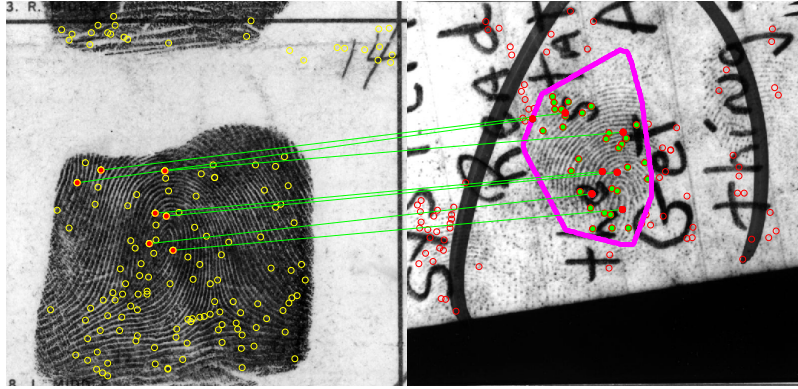
- *Step 4:* Apply crossover operator and mutation operator to the fitter chromosomes, then create the new chromosomes for next generation;
- *Step 5:* Check whether the maximal iterations is reached, or check whether the highest fitness values are maintained the same during several iterations. If not reached or maintained, go back to *Step 3*. Otherwise terminate GA and output the fittest chromosome as the estimated optimal parameter for affine transformation;
- *Step 6:* Align minutiae set  $C$  versus minutiae set  $L$  based on the obtained transformation parameter, then seek for the corresponding minutiae point pairs according to the criterions in Equation (11) (shown in Figure 9(a));
- *Step 7:* Output the number of paired minutiae between  $C$  and  $L$  as the matching score when it has converged as shown in Figure 9(b).

## 3 Experimental Results

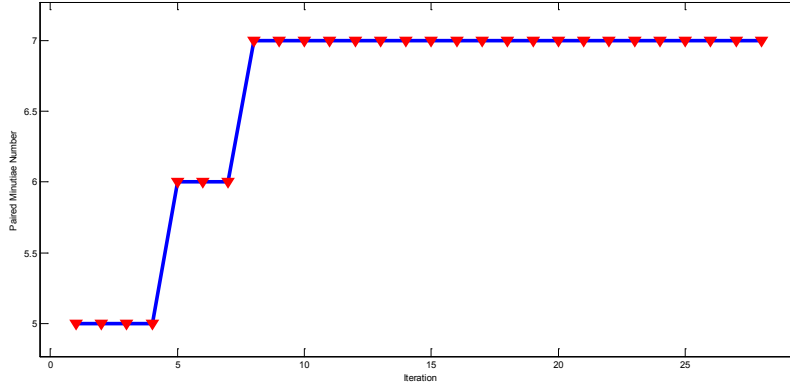
In this section, the introduced latent fingerprint matcher has been evaluated by the following two experiments: (i) the minutiae extraction based on the ROI; and (ii) the latent fingerprint matching.

### 3.1 Data Preparation

All the experiments are conducted on the latent fingerprint database NIST SD27 which is available in the public domain. Such database includes 258 latent print images and their corresponding rolled print images, where these 258 latent images are grouped by the latent



(a)



(b)

Figure 9: The example of successful matching for genuine “rolled-latent” pair by proposed GA-based matching unit. In this case, seven minutiae pairs are found. The paired minutiae between the rolled and latent print are connected by green lines and marked by red solid points respectively. (a) the rolled print image (left) corresponds to G009 in NIST SD27 (right); the automatically extracted minutiae in rolled print image are marked by yellow hollow points and the ones in latent print image are marked by red hollow points; the ROI is highlighted by pink convex polygon and the automatically extracted minutiae within the ROI are marked by green solid points; and (b) the augment of the paired minutiae number with the iteration of GA.

examiners into the following three categories: “Good”, “Bad” and “Ugly”. The numbers of latent images involved in “Good”, “Bad” and “Ugly” categories are 88, 85 and 85 respectively.

### 3.2 Experiment 1: ROI-Based Minutiae Extraction

In this experiment, the reliability of automated minutiae extraction based on the ROI in latent print is evaluated. All the minutiae within the ROI are automatically extracted by VeriFinger SDK. Given query latent print image, the following three scenarios are compared.

- **Minutiae Extraction Scenario 1 - Whole Image-Based Automated Minutiae**

**Extraction:** instead of the segmentation for the foreground, the whole latent image is directly used for the automated minutiae extraction.

- **Minutiae Extraction Scenario 2 - Proposed ROI Segmentation Module-**

**Based Automated Minutiae Extraction:** the ROI is obtained by the proposed dictionary learning-based ROI segmentation module and then used for the automated minutiae extraction. The parameters used in the proposed ROI segmentation module are empirically tuned as follows: the patch size for training sample selection  $w = 32$ , the atom number  $N_a = 100$ , the sparsity parameter  $K = 2$ , and the cross-correlation threshold to judge the pattern similarity  $Th_{xcorr} = 0.6$ .

- **Minutiae Extraction Scenario 3 - Cao’s ROI Segmentation Approach-Based**

**Automated Minutiae Extraction:** the ROI is obtained by [9] and then used for the automated minutiae extraction. The parameters used in this method are tuned as the suggested ones.



In order to assess the performance of the ROI-based automated minutiae extraction, the manually marked minutiae provided by NIST SD27 for query latent fingerprint are used as the ground-truth to label the genuine minutiae. To be more specific,  $MS_1$  is the ground-truth minutiae set (provided by NIST SD27 and marked by the FBI latent fingerprint examiners);  $MS_2$  is the minutiae set automatically extracted from the whole latent image by VeriFinger SDK; and  $MS_3$  is the minutiae set automatically extracted from the ROI by VeriFinger SDK. Accordingly, two metrics such as genuine minutiae preservation rate (GMPR) and false minutiae acceptance rate (FMAR) are defined as follows

$$GMPR = \frac{\# \{MS_1 \cap MS_3\}}{\# \{MS_1 \cap MS_2\}} \quad (12)$$

$$FMAR = \frac{\# \{MS_3 - MS_1 \cap MS_3\}}{\# \{MS_2 - MS_1 \cap MS_2\}} \quad (13)$$

where GMPR is defined as the percentage of the minutiae belonging to the genuine minutiae set which are also correctly extracted by computer program. FMAR is defined as the percentage of the minutiae belonging to the structured noise in latent image which are wrongly detected as the genuine ones by computer program.  $\# \{(*)\}$  denotes the number of minutiae included in set  $(*)$ . Under the condition of fully automated minutiae extraction, the effect of ROI is evaluated depending on the benchmark where no ROI mask is adopted but the whole query latent image is used for automated minutiae extraction. Therefore, in Equations (12) and (13), not the manually marked genuine minutiae  $\# \{MS_1\}$  but the automatically extracted genuine minutiae  $\# \{MS_1 \cap MS_2\}$  and  $\# \{MS_2 - MS_1 \cap MS_2\}$  are used as the baseline. For Scenario 1,  $MS_1 - MS_1 \cap MS_2$  stands for the missing genuine minutiae, even

the entire latent image is imported into the automated minutiae extractor;  $MS_1 \cap MS_2$  represents the genuine minutiae which are correctly detected by the computer program; and  $MS_2 - MS_1 \cap MS_2$  are the spurious minutiae falsely extracted by the computer program. For Scenario 2 and 3,  $MS_1 - MS_1 \cap MS_3$  denotes the missing genuine minutiae after the adoption of ROI;  $MS_1 \cap MS_3$  stands for the genuine minutiae correctly detected by the computer program inside ROI domain; and  $MS_3 - MS_1 \cap MS_3$  represents the false minutiae wrongly extracted by the computer program within ROI.

The minutiae extracted by the automated computer program could not be exactly the genuine ones and unavoidably contain fakes. That is, image-based minutiae detection and extraction is primarily depending on the locally salient image structures such as ridge ending or ridge bifurcation in image domain. For the latent image, such typical structures are not distinct or even lost due to the low clarity of “ridge-valley” pattern in fingerprint region, and consequently lead to the loss of genuine minutiae; some other image components in background also have the similar ridge ending or bifurcation structures, and accordingly yield the spurious minutiae. Therefore, the missing detection for the genuine minutiae and the false detection for the imposter ones by the automated computer program (e.g. VeriFinger SDK) are usually unavoidable. As a consequence, for Scenario 1, the sets  $MS_1 - MS_1 \cap MS_2$  and  $MS_2 - MS_1 \cap MS_2$  are not empty. Similarly, for Scenario 2 and 3,  $MS_1 - MS_1 \cap MS_3$  and  $MS_3 - MS_1 \cap MS_3$  are not empty either.

Based on the three scenarios, their corresponding GMPRs and FMARs on 258 latent images in NIST SD27 are calculated according to the Equation (12) and (13) respectively (shown in Figure 10). Therefore, the mean GMPR ( $\overline{GMPR}$ ) and the mean FMAR ( $\overline{FMAR}$ ) are summarized in Table 1. For the performance comparison among these three scenarios, the

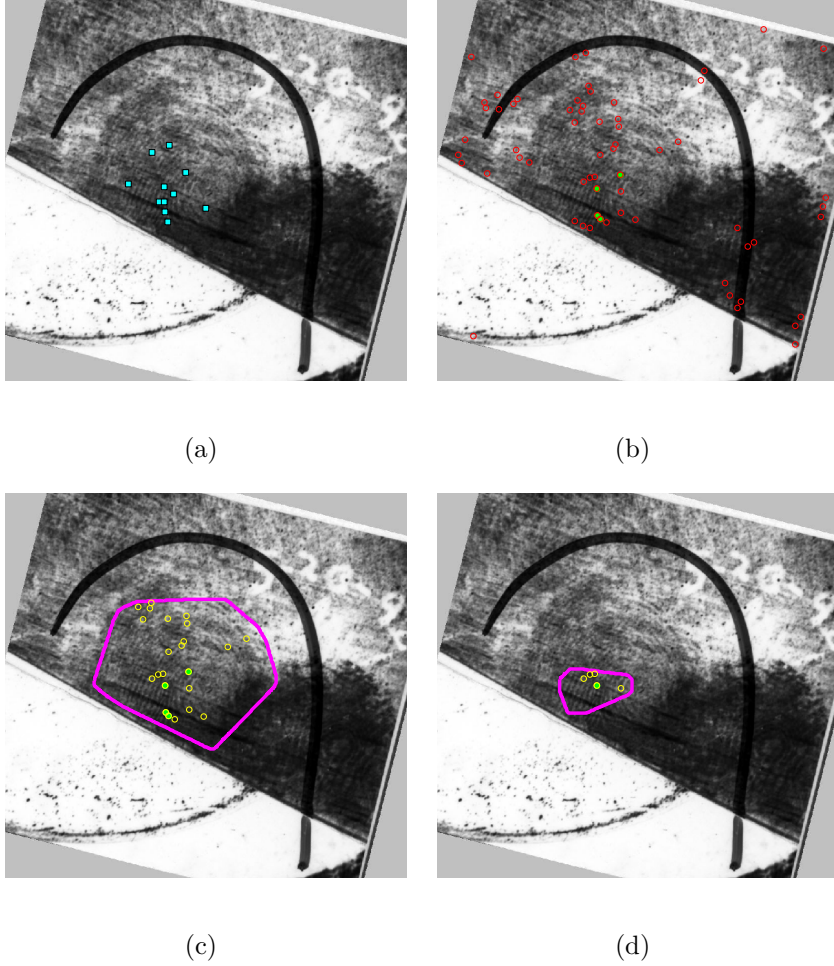


Figure 10: The example of three scenarios on U267 in NIST SD27: (a) the ground-truth minutiae set provided by NIST SD27; (b) Scenario 1 - four genuine minutiae can be correctly extracted,  $GMPR = 1$ ,  $FMAR = 1$ , and  $AUC = 0.5$ ; (c) Scenario 2 - four genuine minutiae can be correctly extracted,  $GMPR = 1$ ,  $FMAR = 0.3167$ , and  $AUC = 0.8416$ ; and (d) Scenario 3 - Only one genuine minutiae can be correctly detected,  $GMPR = 0.25$ ,  $FMAR = 0.0667$ , and  $AUC = 0.5916$ . The red / yellow hollow points stand for the automatically extracted minutiae and the green solid points represent the genuine minutiae labeled by ground-truth in (a). The ROIs in (c) and (d) are obtained by proposed segmentation module and the method reported in [9] and highlighted by pink convex polygons respectively.

Table 1: The comparison of  $\overline{GMPR}$ ,  $\overline{FMAR}$  and  $\overline{AUC}$  among three scenarios

	$\overline{GMPR}$	$\overline{FMAR}$	$\overline{AUC}$
Scenario 1	0.9689	1	0.4845
Scenario 2	0.7772	0.2711	0.7531
Scenario 3	0.6854	0.2642	0.7106

area under the curve (AUC) in receiver operating characteristics analysis (ROC analysis) is adopted. Because the GMPR and the FMAR have the equivalent principle as the true positive rate (TPR: sensitivity) and the false positive rate (FPR: specificity) in ROC analysis, the AUC is used to evaluate the impact of ROI-based automated minutiae extraction. That is, the higher AUC, the more reliable the ROI-based automated minutiae extraction performs. For the statistics of the performance regarding to the three scenarios, their corresponding mean AUCs ( $\overline{AUC}$ ) on overall 258 latent images are calculated. As shown in Table 1, the automated minutiae extraction based on the ROI obtained by the proposed segmentation module achieves the best performance, where the genuine minutiae are effectively preserved while the imposter ones are significantly reduced. In addition, the automated minutiae extractor can not detect even one genuine minutiae for some latent images, although the images in such failed cases are completely imported into the automated minutiae extractor. Accordingly, such failed cases where  $MS_1 \cap MS_2 = \emptyset$  have been evidenced by  $\overline{GMPR} = 0.9689 < 1$  in Table 1.

### 3.3 Experiment 2: Latent Fingerprint Matching

In this session, the matching experiment is implemented to verify whether the proposed dictionary learning-based ROI segmentation module and the GA-based matching unit are able to boost the matching performance. As the ultimate goal of the fully automated latent fingerprint matching, the matching accuracy plays a vital role to demonstrate the improvement caused by the proposed ROI segmentation module and the GA-based matching unit.

In this experiment, the 258 latent images provided by NIST SD27 and a background database are used. Such background database are made up of the 258 mated rolled prints from NIST SD27 and the last 2000 rolled impressions from NIST SD14. Thus, the size of background database is 2258. For each query latent print, it is matched against a small-size subset from the established background database. The subset is generated based on the background database and its size is denoted as  $R$  ( $R < 2258$ ). In such size- $R$  subset, the  $R - 1$  rolled prints are randomly and non-repeatedly selected from the background database, while the remaining one is the mate corresponding to the query latent print (the mate can not be included in the pre-selected  $R - 1$  prints). Considering the intensive computation involved in the iteration of GA, the subset size  $R$  is fixed as 50 and the parallel computation based on a distributed computers system is utilized. For generalizing to the overall 2258 rolled prints in the background database, the matching experiment is conducted 10 times. Besides, the minutiae for the rolled prints and the ROI in latent prints are automatically extracted by VeriFinger SDK. The parameters used in the proposed GA-based matching unit are empirically tuned as follows: the size of chromosomes population  $S = 400$ , crossover probability  $p_c = 0.2$ , mutation probability  $p_m = 0.05$ , minutiae coordinate distance tolerance  $\delta_d = 15$ ,

and minutiae orientation difference tolerance  $\delta_o = 20^\circ$ . Further, the value ranges for the affine transformation parameters are set as follows: the orientation range  $0 \leq \theta \leq 359^\circ$ , the scaling range  $0.8 \leq s \leq 1.2$ , the horizontal shift range  $-400 \leq t_x \leq 400$ , and the vertical shift range  $-400 \leq t_y \leq 400$ .

In order to evaluate and compare the matching performance, the cumulative match characteristic curve (CMC) is utilized. The CMC indicates the identification rate against the penetration rate  $pr$  (e.g.  $pr = 1\%, 5\%, 10\%, 15\%, \dots, 100\%$ ). For example, given a query latent print in this experiment, it is searched against 50 rolled prints. Therefore, the length of candidate list is 50 and  $pr = 10\%$  indicates that the mated rolled print corresponding to the query latent print appears at the  $5^{th}$  ( $5 = 50 \times 10\%$ ) of the candidate list. Further, for the statistics of matching performance, the mean CMC ( $\overline{CMC}$ ) and mean penetration rate  $\overline{pr}$  on overall 258 latent prints during 10 trials are obtained. For the following three scenarios, their corresponding  $\overline{CMC}$  and  $\overline{pr}$  are demonstrated in Figure 11 and Table 2 respectively.

- **Matching Scenario 1 - Proposed GA-Based Matching Unit Only:** without the segmentation module, the minutiae automatically extracted from the whole query latent image are directly imported into the proposed GA-based matching unit.
- **Matching Scenario 2 - Proposed ROI Segmentation Module + Proposed GA-Based Matching Unit:** with the preprocessing performed by the proposed ROI segmentation module, the ROI is obtained; then the minutiae automatically extracted from the obtained ROI are imported into the proposed GA-based matching unit.
- **Matching Scenario 3 - Cao's ROI Segmentation Method + Proposed GA-Based Matching Unit:** by adopting the approach introduced in [9], the ROI is

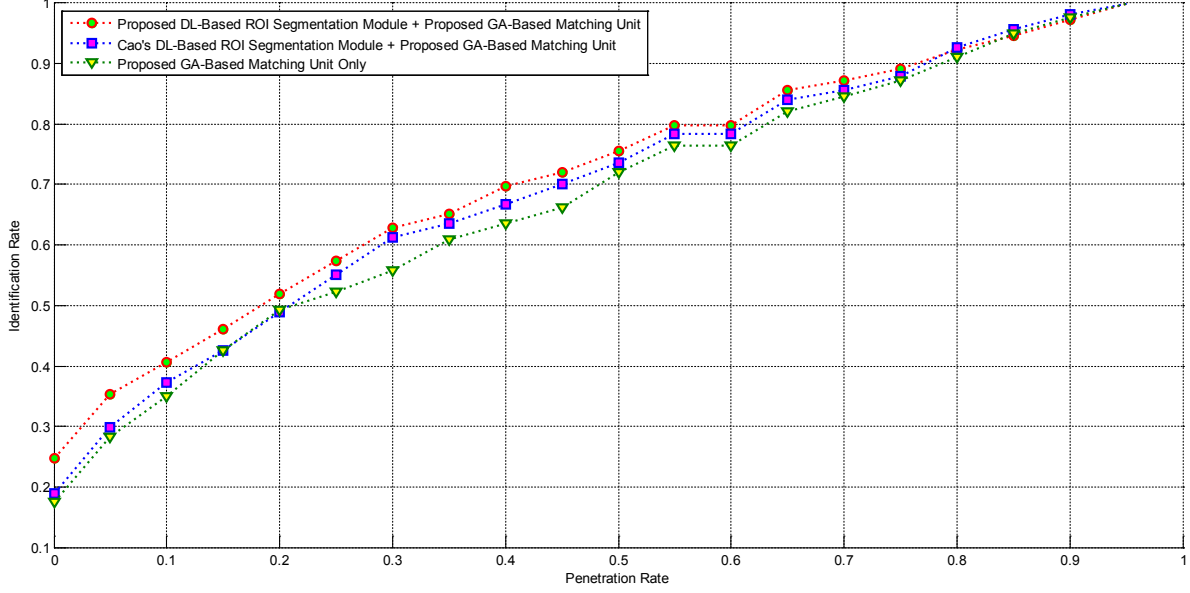


Figure 11: Different  $\overline{CMC}$  obtained by the three scenarios.

Table 2: Different  $\overline{pr}$  obtained by the three scenarios

	Scenario 1	Scenario 2	Scenario 3
$\overline{pr}$	38.159%	34.496%	36.434%

obtained; then the minutiae automatically extracted from the obtained ROI are imported into the proposed GA-based matching unit.

As illustrated in Figure 11, the proposed GA-based matching unit is performed as the baseline latent matcher in Scenario 1. Based on such baseline matcher, the further improvement is expected to be achieved via the ROI segmentation method. Scenario 2 and 3 provide the two different segmentation modules for the ROI identification in latent images respectively. As demonstrated in preceding experiment, the reliability of automated minutiae extraction based on the ROI segmentation module in Scenario 2 is better than that of Scenario 3. As a

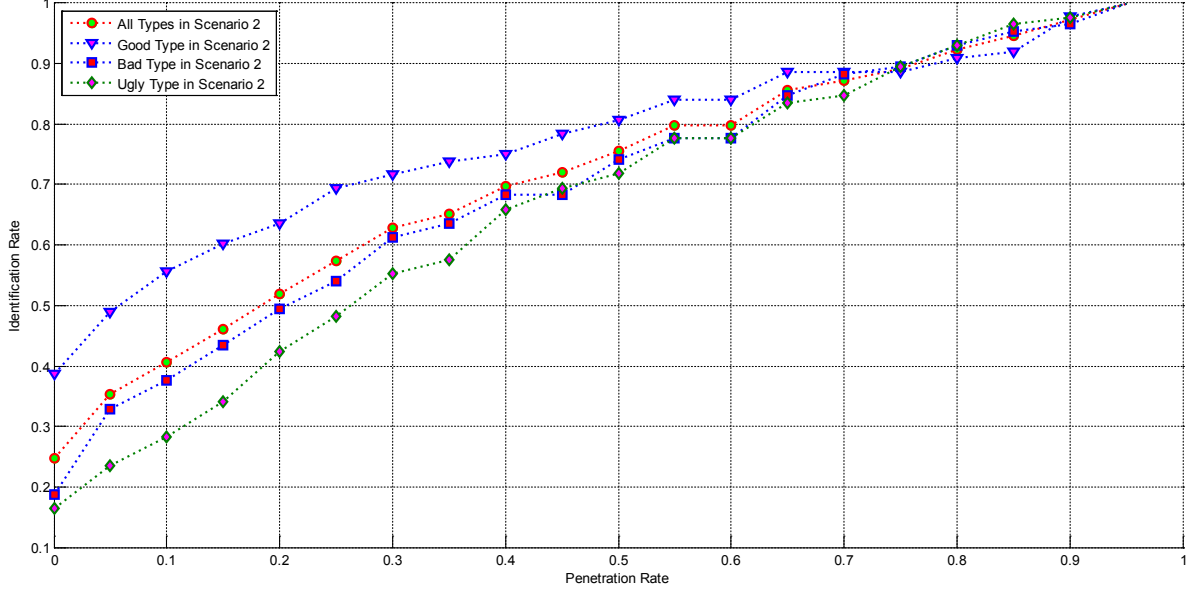


Figure 12:  $\overline{CMC}$  obtained in Scenario 2 according to the three categories of latent images: Good, Bad and Ugly.

result, the matching performance achieved by Scenario 2 is consequently better than that of Scenario 3. The significant decrease with respect to the mean penetration rate  $\overline{pr}$  in Table 2 also demonstrates the effectiveness of the proposed multi-module latent matcher in Scenario 2.

The matching performance on three categories of latent images is further evaluated by Scenario 2. The results shown in Figure 12 and Table 3 demonstrate that  $\overline{CMC}$  and  $\overline{pr}$  achieved based on the “Good” latent images is more satisfying rather than the “Bad” and “Ugly” ones. The “Ugly” ones result in the poor matching performance in terms of  $\overline{CMC}$  and  $\overline{pr}$  because the image quality in “Ugly” subclass is poor indeed.



Table 3:  $\overline{pr}$  obtained in Scenario 2 according to the three categories of latent images: Good, Bad and Ugly

	Good	Bad	Ugly
$\overline{pr}$	28.40%	36.17%	39.11%

## 4 Conclusions and Future Work

Although the impressive matching performance has been achieved in the rolled / plain print identification tasks, the matching for the latent fingerprint is still a challenging problem because of the following issues: (i) the poor-quality image and the low-clarity “ridge-valley” pattern in ROI; and (ii) the corruption with the extra structured image components. Such issues impose the adverse effects on the fully-automated ROI segmentation, minutiae extraction, and minutiae-based matching. In order to deal with the challenges in latent fingerprint matching task, a multi-module latent matching system is introduced in this paper. The proposed latent matcher consists of the following two modules: (i) the dictionary learning-based ROI segmentation scheme; and (ii) the genetic algorithm-based minutiae set matching unit. Experimental results on NIST SD27 latent image database demonstrate the effectiveness of the proposed multi-module latent matching system. In the consideration of its practical value, the proposed matching system can be available in the public domain for latent print identification, and can be also smoothly extended and incorporated with other pre-processing or post-processing modules.

However, the proposed multi-module latent matcher is still not able to handle very low quality latent print images. Since the clarity of the “ridge-valley” pattern in foreground

is poor, the minutiae within such region may not be extracted in a full-automatic mode and consequently the searching against the large-size background database would not be successful. Further, the intensive computation caused by the iteration of GA also needs to be reduced. Therefore, the proposed multi-module latent matching system could be further improved in the following directions:

- The development and integration of the enhancement module for the low-clarity “ridge-valley” pattern in foreground is necessary in the on-going research;
- The intensive computation involved in the iteration of GA-based matching unit could be further reduced by defining the simpler fitness function, or exploring to utilize the large-scale parallel or distributed computing system.

## References

- [1] H. Faulds, On the skin-furrows of the hand, *Nature*, XXII: 605, 1880.
- [2] D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar, *Hand book of fingerprint recognition*, Springer Verlag, 2009.
- [3] P. Komarinski, Automated fingerprint identification systems (AFIS), *Elsevier Academic Press*, 2005.
- [4] Y. Wang, J.K. Hu, Global ridge orientation modelling for partial fingerprint identification, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 72-87, 2011.

- [5] A.K. Jain, J.J. Feng, K. Nandakumar, On matching latent fingerprints, *Proceedings of IEEE Computer Vision and Pattern Recognition Workshop Biometrics*, pp. 1-8, 2008.
- [6] A.K. Jain, J.J. Feng, Latent fingerprint matching, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 88-100, 2011.
- [7] A.A. Paulino, J.J. Feng, A.K. Jain, Latent fingerprint matching using descriptor-based hough transform, *Proceedings of IEEE International Joint Conference on Biometrics Compendium*, pp. 1-7, 2011.
- [8] A.A. Paulino, J.J. Feng, A.K. Jain, Latent fingerprint matching using descriptor-based hough transform, *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 31-45, 2013.
- [9] K. Cao, E.Y. Liu, A.K. Jain, Segmentation and enhancement of latent fingerprints: a coarse to fine ridge structure dictionary, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, 2014.
- [10] J.Y. Zhang, R.J. Lai, C.J. Kuo, Adaptive directional total-variation model for latent fingerprint segmentation, *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, 2013.
- [11] J.Y. Zhang, R.J. Lai, C.J. Kuo, Latent fingerprint detection and segmentation with a directional total variation model, *Proceedings of IEEE International Conference on Image Processing*, pp. 1145-1148, 2012.

- [12] S.K. Ashtiani, C.J. Kuo, A robust technique for latent fingerprint image segmentation and enhancement, *Proceedings of IEEE International Conference on Image Processing*, pp. 1492-1495, 2008.
- [13] H. Choi, M. Boaventura, I.A.G. Boaventura, A.K. Jain, Automatic segmentation of latent fingerprints, *Proceedings of IEEE International Conference on Biometrics Compendium*, pp. 303-310, 2012.
- [14] N.J. Short, M.S. Hsiao, A.L. Abbott, E.A. Fox, Latent fingerprint segmentation using ridge template correlation, *Proceedings of International Conference on Imaging for Crime Detection and Prevention*, pp. 1-6, 2011.
- [15] X.J. Tan, B. Bhanu, Fingerprint matching by genetic algorithms, *Pattern Recognition*, vol. 39, pp. 465-477, 2006.
- [16] K. Engan, S.O. Aase, J.H. Husoy, Multi-frame compression: theory and design, *Signal Processing*, vol. 80, no. 10, pp. 2121-2140, 2000.
- [17] M. Elad, M. Aharon, Image denoising via sparse and redundant representation over learned dictionaries, *IEEE Transactions on Image Processing*, vol. 15, pp. 3736-3745, 2006.
- [18] I. Todic, P. Frossard, Dictionary learning, *IEEE Signal Processing Magazine*, vol. 28, pp. 27-38, 2011.
- [19] J. Mairal, F. Bach, J. Ponce, G. Sapiro, Online dictionary learning for sparse coding, *Proceedings of International Conference on Machine Learning*, pp. 689-696, 2009.

- [20] J. Tropp, Greed is good: Algorithmic results for sparse approximation, *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2231-2242, 2004.
- [21] E. Agoston, S. James, Introduction to evolutionary computing, *Springer*, 2003.
- [22] B.H. Li, Q.G. Meng, H. Holstein, Point pattern matching and applications - a review, *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, vol. 1, pp. 729-736, 2003.
- [23] L.H. Zhang, W.L. Xu, C. Chang, Genetic algorithm for affine point pattern matching, *Pattern Recognition Letters*, vol. 24, pp. 9-19, 2003.
- [24] J.W. Xu, J.K. Hu, X.P. Jia, Genetic algorithm for distorted point set matching, *Proceedings of International Congress on Image and Signal Processing*, vol. 3, pp. 1724-1729, 2013.
- [25] N. Otsu, A threshold selection method from gray-level histograms, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62-66, 1979.